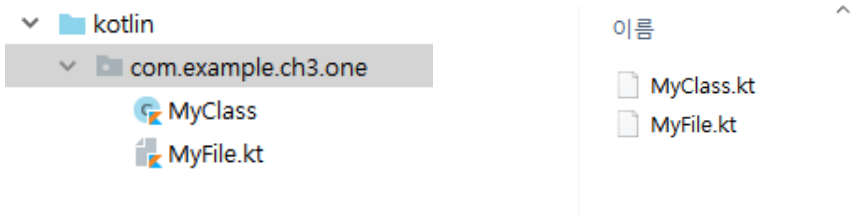
3장. 코틀린 기본 작성법

3.1. 코틀린 파일 정의

3.1.1. 일반 파일과 클래스 파일

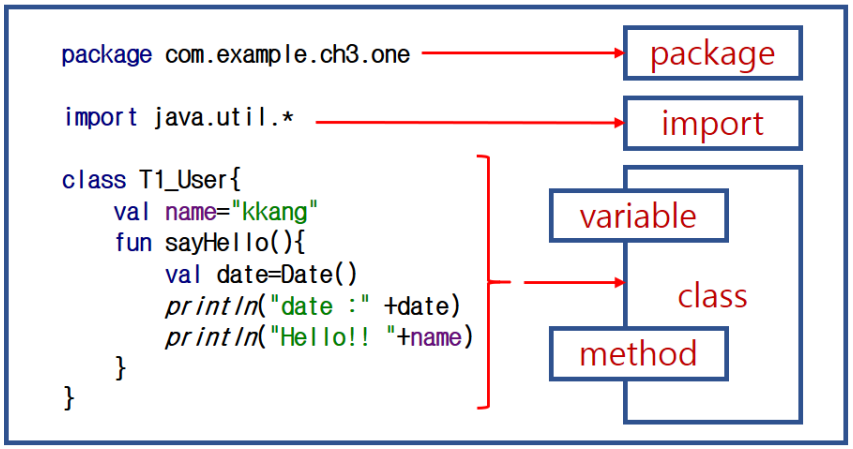
- 코틀린 프로그램은 확장자가 kt인 파일
- 파일(File)과 클래스 파일(Class) 편의상 구분



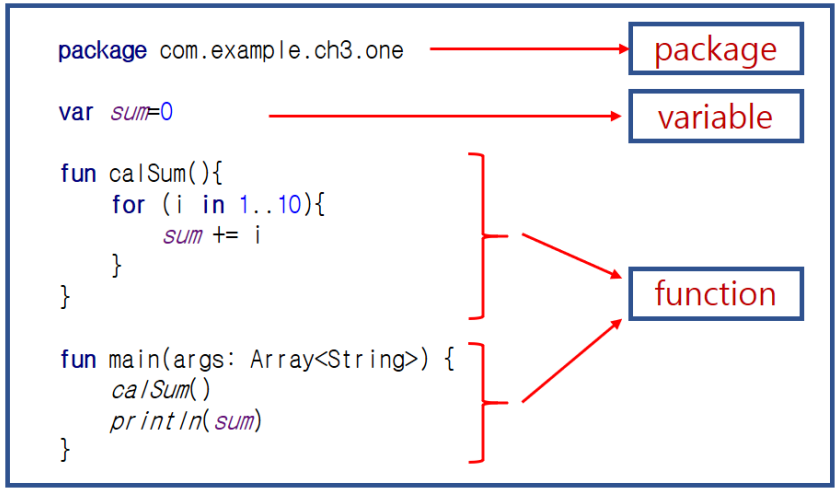
3.1. 코틀린 파일 정의

3.1.2. 파일의 구성요소

- 패키지(package), 임포트(import), 클래스, 변수, 함수 선언과 주석이 파일에 포함

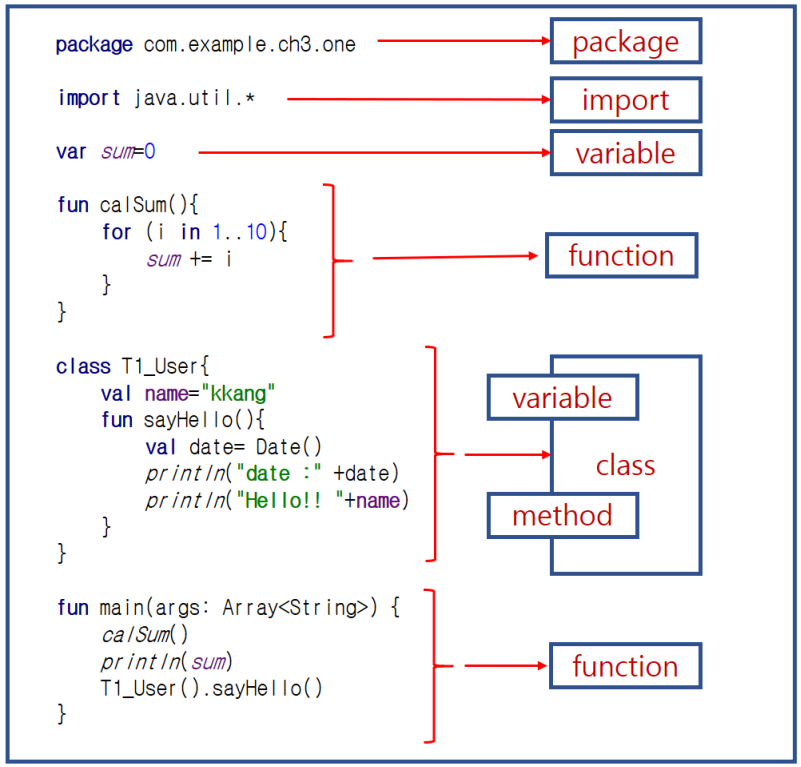


- 클래스를 사용하지 않고 변수와 함수로만 구성가능



3.1. 코틀린 파일 정의

- 코틀린 파일에 패키지, 임포트, 변수, 함수, 클래스 등을 모두 선언

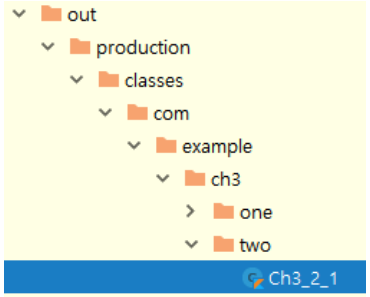


3.2. 패키지

3.2.1. 패키지 기본 개념

- 패키지(package)란 관련된 클래스들을 묶기 위한 물리적인 개념

```
package com.example.ch3.two

class Ch3_2_1
```



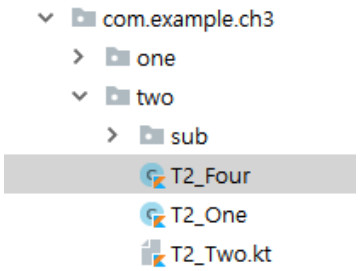
- 이용하려는 클래스가 다른 패키지에 있다면 import 구문

```
package com.example.ch3.two

import com.example.ch3.two.sub.T2_Three

fun main(args: Array<String>) {
    val one=T2_One()
    val three=T2_Three()
}
```

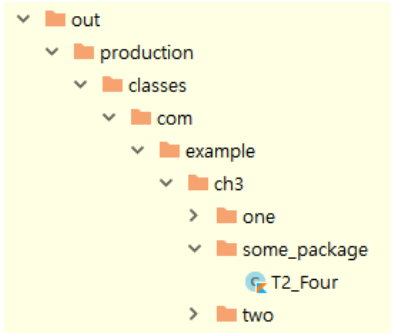
3.2. 패키지

3.2.2. 가상 패키지

- 코틀린 파일이 있는 폴더와 다른 패키지명을 사용할 수 있다




```
Package com.example.ch3.some_package

class T2_Four
```



3.2. 패키지

3.2.3. 변수/함수 임포트

- 클래스로 묶지 않은 변수와 함수를 최상위 레벨로 관리
- 패키지 내에 선언된 전역변수나 전역함수처럼 취급

```
package com.example.ch3.two.sub

val threeVal=10

fun threeFun(){

}
```

```
package com.example.ch3.two

import com.example.ch3.two.sub.threeFun
import com.example.ch3.two.sub.threeVal

fun main(args: Array<String>) {
    println(threeVal)
    threeFun()
}
```

3.2. 패키지

3.2.4. 기본 패키지

- kotlin.*
- kotlin.annotation.*
- kotlin.collections.*
- kotlin.comparisons.* (since 1.1)
- kotlin.io.*
- kotlin.ranges.*
- kotlin.sequences.*
- kotlin.text.*
- java.lang.*
- kotlin.jvm.*

3.2. 패키지

3.2.5. 이름 변경해서 임포트하기

- 임포트할 때 이름을 바꾸어 다른 이름으로 사용가능.

```
import java.text.SimpleDateFormat as MySimpleDateFormat
```